Collaborative Filtering Movie Recommender System

Hongbin Qu
University of Washington

Abstract

Recommender systems play an important role in the information era. It can be applied in various aspects such as music apps, movie websites, social media, etc. Collaborative filtering is a common approach to design a recommender system. This paper discusses several ways to implement memory-based collaborative filtering on Movielens dataset and evaluates their performance.

Introduction

There are generally three recommendation types: collaborative filtering, content-based filtering, and hybrid recommender system (the combination of the first two approaches). The core idea of collaborative filtering algorithms is the similarity of two users' past preference can predict a user's future preference. In this project, couples of collaborative filtering algorithm are performed. The data will be analyzed is MovieLens datasets, which is a rating dataset contains 100,000 ratings and 3,600 tag applications applied to 9,000 movies by 600 users

User-Based and Item-Based CF

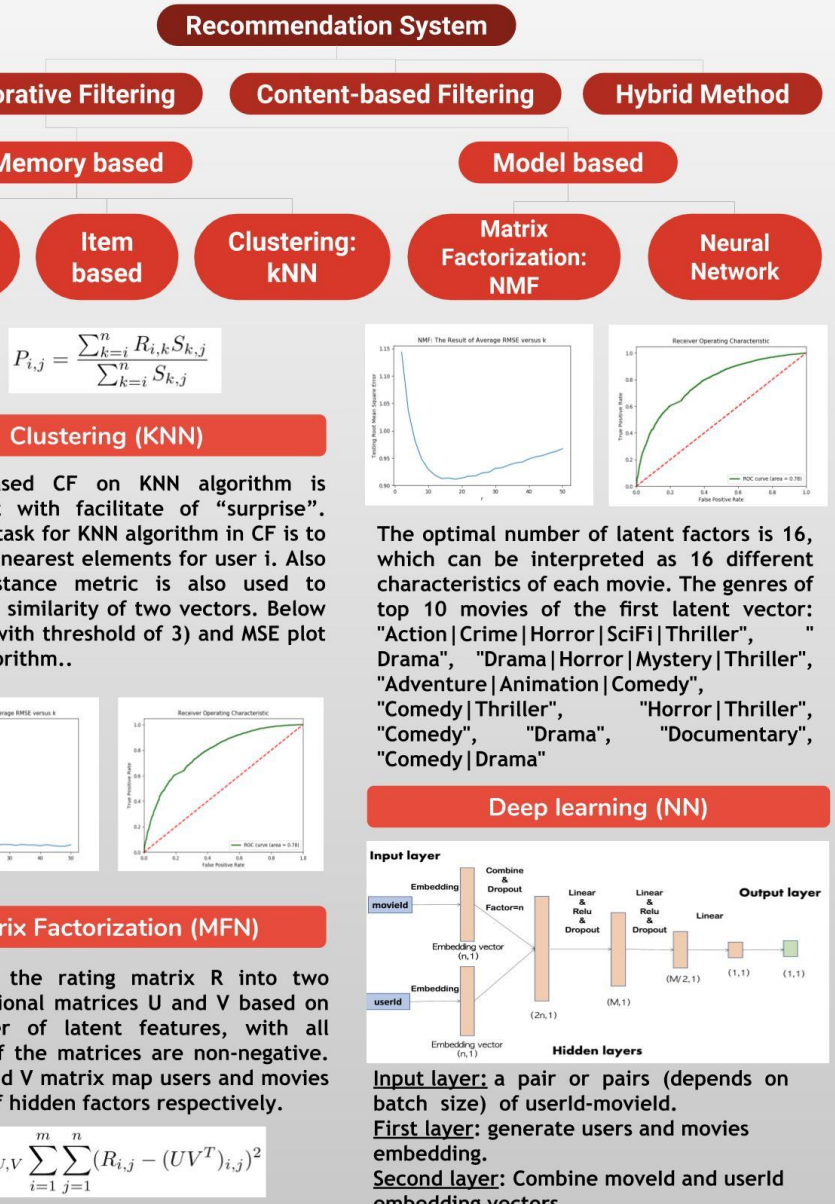
Given rating matrix R, number of users n, number of movies m.
Cosine distance is used to measure the distance between two vector:

$$sim(x,y) = \frac{x \cdot y}{||x|| \times ||y||} = \sum_i \frac{x_i y_i}{\sqrt{\sum_i x_i^2}\sqrt{\sum_i y_i^2}}$$

For user-based approach: similarity matrix is a m × m matrix, the prediction is:

$$P_{i,j} = \overline{R}_i + \frac{\sum_{k=i}^{n} S_{i,k}(R_{k,j} - \overline{R}_k)}{\sum_{k=i}^{n} S_{i,k}}$$

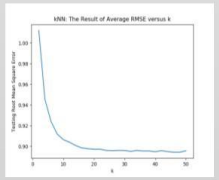
For item-based approach: similarity matrix is a n × n matrix, the prediction is:



$$P_{i,j} = \frac{\sum_{k=i}^{n} R_{i,k} S_{k,j}}{\sum_{k=i}^{n} S_{k,j}}$$

Clustering (KNN)

A user-based CF on KNN algorithm is implement with facilitate of "surprise". The main task for KNN algorithm in CF is to find the k nearest elements for user i. Also cosine distance metric is also used to define the similarity of two vectors. Below are ROC (with threshold of 3) and MSE plot of this algorithm..



Matrix Factorization (MFN)

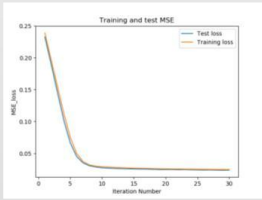
Factorizing the rating matrix R into two low-dimensional matrices U and V based on the number of latent features, with all elements of the matrices are non-negative. U matrix and V matrix map users and movies into a set of hidden factors respectively.

$$min_{U,V} \sum_{i=1}^{m} \sum_{j=1}^{n} (R_{i,j} - (UV^T)_{i,j})^2$$

The optimal number of latent factors is 16, which can be interpreted as 16 different characteristics of each movie. The genres of top 10 movies of the first latent vector: "Action|Crime|Horror|SciFi|Thriller", "Drama", "Drama|Horror|Mystery|Thriller", "Adventure|Animation|Comedy", "Comedy|Thriller", "Horror|Thriller", "Comedy", "Drama", "Documentary", "Comedy|Drama"

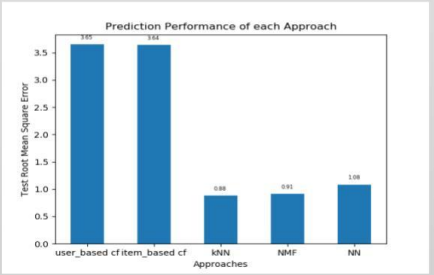
Deep learning (NN)

Input layer: a pair or pairs (depends on batch size) of userId-movieId.
First layer: generate users and movies embedding.
Second layer: Combine moveId and userId embedding vectors
Third, Fourth and Fifth layer: Linear network layers
Output layer: Prediction result, scalar.
Loss Function & Optimizer: MSELoss, SGD
Hyperparameter: Learning Rate (lr), momentum, number of factors(embedding layer), outputs size in the third layer (M).
Process and Results: The experiment was performed 20 trials, there are 30 epoches in each trial. The optimal hyperparameters: lr=1e-5, momentum=0.4, M=42 and embedding factor is 50. Minimum training loss is 0.023, minimum test losss is 0.024 and minimum test error is 1.08.



Conclusion

To summarize, the root mean square error of each approach is shown below:



Rererence

[1] Patel, Khagesh, and Ankush Sachdeva. "Hybrid Recommendation System."(2014)

[2] F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. ACM Transactions on Interactive Intelligent Systems (TiiS) 5, 4, Article 19 (December 2015), 19 pages. DOI=http://dx.doi.org/10.1145/2827872

[3] Schafer, J. Ben, et al. "Collaborative filtering recommender systems." The adaptive web. Springer, Berlin, Heidelberg, 2007. 291-324.

[4] Welcome to Surprise' documentation!. (n.d.). Retrieved from https://surprise.readthedocs.io/en/stable/index.html

[5] Lee, Daniel D., and H. Sebastian Seung. "Learning the parts of objects by non-negative matrix factorization." Nature 401.6755 (1999): 788.

[6] Shioulin."PyTorch for Recommenders 101", 10 April 2018, https://blog.fastforwardlabs.com/2018/04/10/pytorch-for-recommenders-101.html

[7] EthanRosenthal."torchmf", 18 May 2017. https://github.com/EthanRosenthal/torchmf/blob/master/torchmf.py